

# MPhil in Advanced Computer Science

## Denotational Semantics

**Leader:** Marcelo Fiore (course lecturer)  
**Timing:** Michaelmas  
**Prerequisites:** Basic computer science and mathematical background  
**Structure:** 8 lectures + 4 exercise classes

### AIMS

This module aims to introduce domain theory and denotational semantics, and to show how they provide a mathematical basis for reasoning about the behaviour of programming languages.

### SYLLABUS

- **Introduction.** The denotational approach to the semantics of programming languages. Recursively defined objects as limits of successive approximations.
- **Least fixed points.** Complete partial orders (cpo) and least elements. Continuous functions and least fixed points.
- **Constructions on domains.** Flat domains. Product domains. Function domains.
- **Scott induction.** Chain-closed and admissible subsets of cpo and domains. Scott's fixed-point induction principle.
- **PCF.** The Scott-Plotkin language PCF. Evaluation. Contextual equivalence.
- **Denotational semantics of PCF.** Denotation of types and terms. Compositionality. Soundness with respect to evaluation.
- **Relating denotational and operational semantics.** Formal approximation relation and its fundamental property. Computational adequacy of the PCF denotational semantics with respect to evaluation. Extensionality properties of contextual equivalence.
- **Full abstraction.** Failure of full abstraction for the domain model. PCF with parallel or.

### OBJECTIVES

At the end of the course students should

- be familiar with basic domain theory: cpo, continuous functions, admissible subsets, least fixed points, basic constructions on domains;
- be able to give denotational semantics to simple programming languages with simple types;

- be able to apply denotational semantics; in particular, to understand the use of least fixed points to model recursive programs and be able to reason about least fixed points and simple recursive programs using fixed point induction;
- understand the issues concerning the relation between denotational and operational semantics, adequacy and full abstraction, especially with respect to the language PCF.

## **COURSEWORK**

Exercises will be provided.

## **PRACTICAL WORK**

N/A

## **ASSESSMENT**

The course will be assessed by means of a written test to be set and marked by the course lecturer.

## **RECOMMENDED READING**

### **Books**

- [1] C. Gunter. *Semantics of programming languages: Structures and techniques*. MIT Press, 1992.
- [2] G. Winskel. *The formal semantics of programming languages: An introduction*. MIT Press, 1993.
- [3] R. Tennent. *Semantics of programming languages*. Prentice Hall, 1991.

### **Papers**

- [1] M. Fiore, A. Jung, E. Moggi, P. O’Hearn, J. Riecke, G. Rosolini, I. Stark. Domains and denotational semantics: History, accomplishments and open problems. *Bulletin of EATCS*, 59:227–256, 1996.
- [2] C.-H. Ong. Correspondence between operational and denotational semantics. *Handbook of Logic in Computer Science*, Vol. 4, pp. 269–356, 1995.
- [3] G. Plotkin. LCF considered as a programming language. *Theoretical Computer Science*, 5:223–256, 1977.
- [4] D. Scott. A type-theoretical alternative to CUCH, ISWIM, OWHY. Typescript, 1969. (In *Theoretical Computer Science*, 121:411–440, 1993.)

Last updated: July 2010